An Interop Publication

CONNEXIONS®

The Interoperability Report

January 1992 Volume 6, No. 1

ConneXions — The Interoperability Report tracks current and emerging standards and technologies within the computer and communications industry.

In this issue:

From the Editor

Our series *Components of OSI* continues this month with a tutorial on *Abstract Syntax Notation One* (ASN.1). At the application layer, the data structures exchanged by protocol entities are often quite complex. Therefore, OSI introduces a formalism for describing these structures. This formalism, termed an *abstract syntax,* is used to define data without regard to machine-oriented structures and restrictions. Once data structures can be described in a machine-independent fashion, there must also be some way of transmitting those structures, unambiguously, over the network. This is the job of the *transfer syntax.* In OSI, this is called the *Basic Encoding Rules* (BER). Howard Motteler and Deepinder Sidhu from the University of Maryland present an overview of the ASN.1 and BER standards defined by ISO and CCITT.

While TCP/IP cynics may refer to ASN.1 as "Asinine One," and complain about the lack of powerful ASN.1 compilers, it should be noted that the Internet community has in fact adopted a subset of ASN.1 as the standard method for describing Management Information Bases (MIBs) in SNMP. This is perhaps the first example of OSI technology being embraced by the Internet, but it will certainly not be the last time this happens as OSI technology reaches maturity.

Réseaux Associés pour la Recherche Européenne (RARE) is the association of European networking organizations and their users. The aim of RARE is to foster cooperation between both national and international networking organizations in order to develop a harmonized data communications infrastructure. To achieve this aim, RARE supports the principles of open systems as defined by ISO and the work on functional profiles undertaken by the *European Workshop for Open Systems* (EWOS) and the *European Telecommunications Standards Institute* (ETSI). We asked the RARE Secretariat to give us an overview of this important organization.

A number of initiatives are currently underway within the DoD to chart aspects of the transition process towards the use of OSI/GOSIP applications and communication protocols. Bob Cooney of the Naval Computer and Telecommunications Station in Washington, DC, outlines some of these efforts.

Also in this issue you will find a book review, information on how to obtain electronic versions of RFC documents, and a couple of workshop announcements. Included with this month's issue is the 1991 index sheet, as well as bound index sheets from previous years (1987–1990). All back issues are available for purchase either as complete volumes or individual copies.

ConneXions is published monthly by Interop, Inc., 480 San Antonio Road, Suite 100, Mountain View, CA 94040, USA. 415-941-3399. Fax: 415-949-1779. Toll-free: 1-800-INTEROP.

ISSN 0894–5926

Components of OSI: Abstract Syntax Notation One (ASN.1)

by
Howard Motteler and Deepinder Sidhu,
University of Maryland

Abstract

Distributed applications often involve exchanges of values of complex data types. There is a need for an unambiguous and machine-independent definition of data types and rules for unambiguous representation of data values for transmission across the network. The International Standards for *Abstract Syntax Notation One* (ASN.1) and ASN.1 *Basic Encoding Rules* (BER) are defined for this purpose. This article presents an overview of the ASN.1 and BER standards defined by ISO and CCITT.

Introduction

Data communication between computers requires solving several basic and fundamental problems. Some of these problems are related to the heterogeneous nature of computers manufactured by different vendors. For example, two machines that need to communicate may use different conventions for representing characters, or different byte orders for integers. At the application layer, communication may involve exchanges of complex data values which are represented independently in the environment of the communicating machines.

Several considerations at the application layer necessitate an external data representation language which is independent of programming languages, operating systems and machine architectures. Such a language must have a formal notation for specifying abstract data types and rules for unambiguous representation of data values for communication on serial lines. ASN.1 is such a formal language. It consists of two parts—*abstract syntax* for the specification of abstract data types, and *transfer syntax,* which gives encoding rules. ASN.1 has a rich syntax for describing data types. It also provides some powerful features such as *tags* for defining new types and *macros* for creating and using extended notation for types.

The ISO Standard 8824 [1], Abstract Syntax Notation One (ASN.1), describes an abstract syntax for specifying data types in a machine-independent manner. The ISO Standard 8825 [2], Basic Encoding Rules for Abstract Syntax Notation One (ASN.1), describes the transfer syntax for unambiguous representation of data values for transmission over the network. The CCITT Recommendations X.205 [3] and X.209 [4] are related CCITT documents comparable to ISO standards 8824 and 8825. Extensions to ASN.1 syntax and encoding rules are discussed in [5,6]. This article presents an overview of ASN.1 and its Basic Encoding Rules. For other discussions of ASN.1, see [6–12].

Abstract syntax

The abstract syntax for ASN.1 data types and values is defined by a set of BNF style productions with alternatives. An example of an ASN.1 production is:

```
BitStringValue ::=      bstring |
                        hstring |
                        {IdentifierList} |
                        { }
```

which says that *BitStringValue* can be any *bstring,* any *hstring,* any sequence associated with *IdentifierList* or an *empty* sequence. These four different alternatives are indicated by three occurrences of symbol | in the right side of this production. A *bstring* consists of an arbitrary number of binary bits (0's and 1's) preceded by symbol ' and followed by a pair of symbols 'B.

Similarly, an *hstring* consists of an arbitrary number of hexadecimal characters preceded by a symbol ' and followed by a pair of symbols 'H. Examples of *bstring* and *hstring* are '10011010101'B and 'D16A052E9'H respectively.

The ASN.1 lexical items consist of sequences of characters from the character set:

A – Z *a – z* 0 – 9 : = , { } < > | . ( ) [ ] – ' "

Additional characters may appear in comments and some string types.

ASN.1 is case-sensitive (upper and lower case letters are distinct). It uses specific case conventions for different categories of names. The first letter of a module, type, local type, local value, or production reference must be an upper case letter. The first letter of a value reference or an identifier must be a lower case letter. The letters in a macro name are all in upper case.

ASN.1 defines several reserved words and keywords. They are summarized in Table 1 on the next page.

Comments are an important part of ASN.1 type or value descriptions. A comment may contain information that enhances readability or it may specify syntactic constraints such as constraint on a parameter range and restriction on a set. A comment begins with a double hyphen "– –" and ends with another double hyphen or the end of the line.

ASN.1 provides several built-in or "primitive" data types similar to built-in types in other programming languages like C and Pascal. It also provides mechanisms to build more complex types from primitive types.

The syntax of ASN.1 type and value declaration is:

NameOfType ::= TYPE
nameOfValue NameOfType ::= VALUE

where *TYPE* is some ASN.1 type, and *VALUE* an ASN.1 value. The first statement declares *NameOfType* as a new type. The second statement assigns a value to variable *nameOfValue* of type *NameOfType*.

Simple types

ASN.1 provides several simple data types as part of the language definition. These types are:

- **BOOLEAN:** one of two distinguished values TRUE or FALSE
- **INTEGER:** an arbitrary precision integer
- **ENUMERATED:** an explicit list of integer values that an instance of a data type may take
- **REAL:** an arbitrary precision real number
- **BITSTRING:** an ordered list of zero or more bit values
- **OCTET STRING:** an ordered list of zero or more octets (sequences of 8 bits)

continued on next page

Abstract Syntax Notation One *(continued)*

Examples of some of these types and value declarations are:

```
Married ::= BOOLEAN
SocialSecurityNumber ::= INTEGER
DayOfTheMonth ::= INTEGER {first(1), last(31)}
                        -- defines the maximum and minimum
                        -- values of an integer type
CloudyDaysOfWeek ::= BIT STRING {first(1), last(7)}
                        -- model the values of a bit map to
                        -- indicate whether a particular condition
                        -- holds such as
                        -- "Day i is cloudy iff bit i is 1"
valMarried Married ::= FALSE
valCloudyDaysOfWeek CloudyDaysOfWeek ::= '1001101'B
firstSundayOfMonth DayOfTheMonth ::= 4
```

Reserved words (ISO 8824)		
ANY	APPLICATION	BEGIN
BIT	BOOLEAN	CHOICE
COMPONENTS	DEFAULT	DEFINITIONS
END	EXTERNAL	FALSE
IDENTIFIER	IMPLICIT	INTEGER
NULL	OBJECT	OCTET
OF	OPTIONAL	PRIVATE
SEQUENCE	SET	STRING
TRUE	UNIVERSAL	
Predefined type references (ISO 8824)		
NumericString	Printable String	TeletexString
T61String	VideotexString	VisibleString
ISO646String	IA5String	GraphicString
GeneralString	GeneralizedString	UTCTime
EXTERNAL	ObjectDescriptor	
Reserved words for macros (ISO 8824)		
MACRO	NOTATION	TYPE
VALUE	type	value
Reserved keywords for symbol elements in macros (ISO 8824)		
empty	identifier	number
string		
Reserved keywords for object identifier values (ISO 8824)		
administration	ccitt	identified-organization
iso	joint-iso-ccitt	member-body
network-operator	question	recommendation
registration-authority	standard	
Reserved words *[Extended ASN.1]*.		
EXPLICIT	ENUMERATED	EXPORTS
IMPORTS	REAL	
Reserved keywords for subtypes *[Extended ASN.1]*.		
ABSENT	BY	COMPONENT
DEFINED	FROM	INCLUDES
MIN	MINUS-INFINITY	MAX
PRESENT	PLUS-INFINITY	SIZE
TAGS	WITH	

Table 1: Reserved words and keywords

Constructor types

ASN.1 provides several *constructors* for building complex data types from simple data types. These constructors and their associated types are:

- **SEQUENCE:** an ordered list of zero or more elements, each of which is a valid ASN.1 type
- **SEQUENCE OF:** same as **SEQUENCE** except that each element is the same ASN.1 type
- **SET:** an unordered list of zero or more elements, each of which is a valid ASN.1 type
- **SET OF:** same as **SET** except that each element is the same ASN.1 type
- **CHOICE:** a union of one or more ASN.1 types

Examples of some of these types and value declarations are:

```
StudentRecord ::= SEQUENCE {
                        identifier      INTEGER,
                        first           OCTET STRING,
                        middle          OCTET STRING OPTIONAL,
                        last            OCTET STRING,
                        married         BOOLEAN
                                        DEFAULT FALSE }
CustomerIdentifier ::= CHOICE {name VisibleString, number INTEGER}
valStudentRecord StudentRecord ::= {
                        identifier      10203040,
                        first           "John",
                        last            "Smith",
                        married         FALSE }
```

There are several noteworthy features of the fields of the constructor type declarations above. The field of a constructor type can be optional which is indicated by the keyword **OPTIONAL**. A field can be assigned default value (during encoding) which is indicated by the ASN.1 value following the keyword **DEFAULT**. A field of a constructor type can be given a name which is indicated by a textual label preceding the type declaration in that field. In the above examples, "identifier," "first," "middle," "last," "married," "name" and "number" are names or textual labels associated with the fields of **SEQUENCE** and **CHOICE** type declarations.

Object types

ASN.1 defines types whose values are used to name information objects such as applicable standards documents. An important use of this is to allow data types to be formally grouped with their documentation. Two such types are:

- **OBJECT IDENTIFIER:** an authoritatively named object
- **OBJECT DESCRIPTOR:** a human-readable, not necessarily unique, textual string that references an **OBJECT IDENTIFIER**

The semantics of an **OBJECT IDENTIFIER** value is defined with respect to a tree structure, the object identifier tree, whose root corresponds to the document defining the ASN.1 standard. The vertices of this tree correspond to the administrative authority responsible for assigning arcs from that vertex. Each information object is uniquely assigned to a vertex, usually a leaf vertex. The arcs of the identifier tree are labeled by object identifier components which are numeric values. An information object is uniquely identified by the sequence of numeric values on the edges along the path from the root of the tree to the vertex identifying the object. The structure of the object identifier tree is shown in Figure 1.

continued on next page

Abstract Syntax Notation One *(continued)*

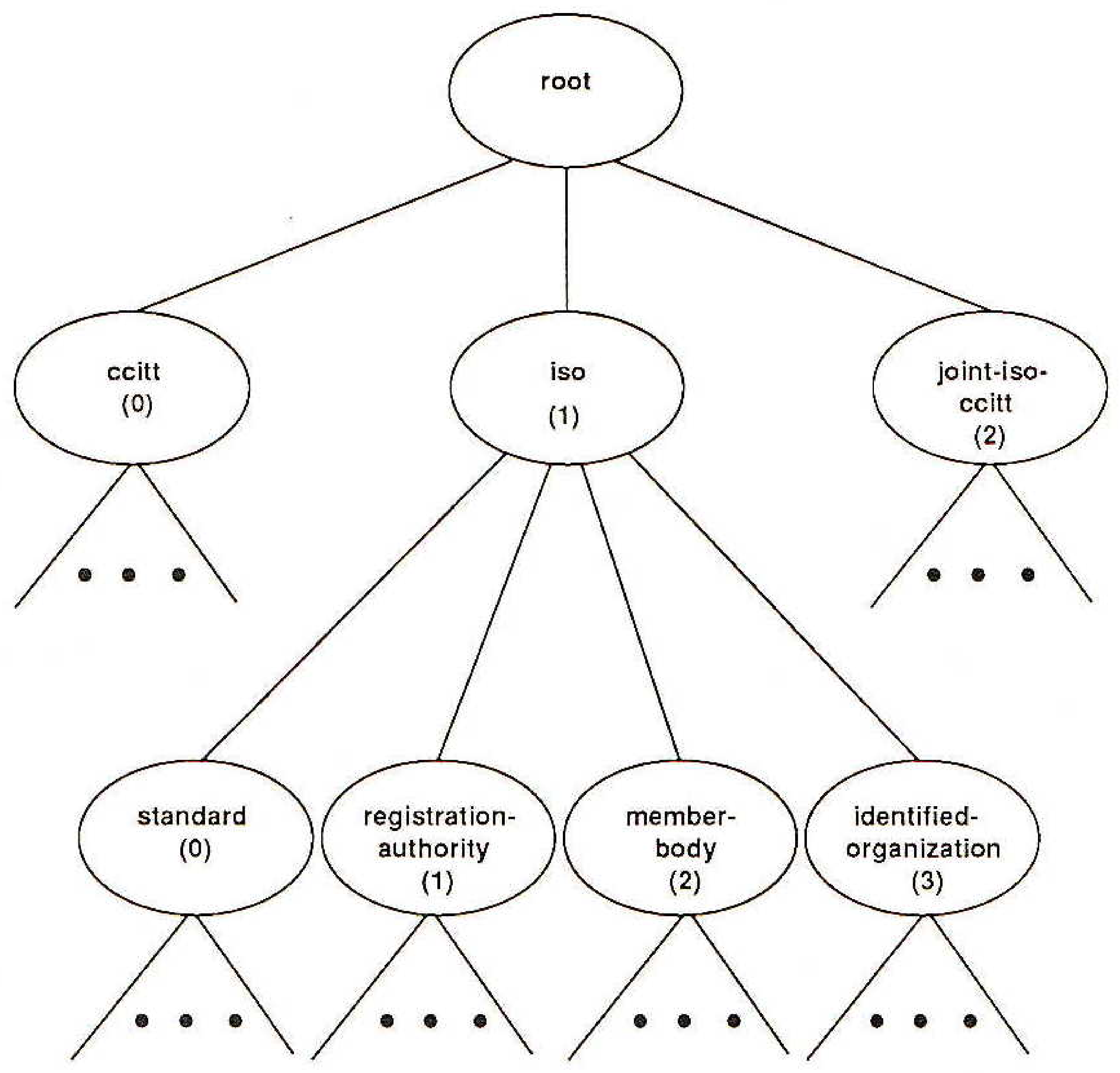

Figure 1: Identifier tree structure

The root has three subordinates which are:

- ccitt (0) — this is administrated by CCITT;
- iso (1) — this is administrated by ISO and IEC;
- joint-iso-ccitt (2) — this is administrated jointly by ISO/IEC and CCITT.

The node labeled iso (1) has four subordinate nodes. They are discussed in [1].

An object identifier value, a sequence of numerical labels, can be represented as a sequence of values enclosed by symbols { and }, e.g.,

{ 1 0 8571 5 1 }

Examples of **OBJECT IDENTIFIER** types and value declarations are:

DocumentTypeName ::= **OBJECT IDENTIFIER**
fTAM-1 DocumentTypeName ::= { 1 0 8571 5 1 }

Character string types

ASN.1 defines several built-in character string types. The values of each of these types are sequences of zero or more characters from a well-defined character set. The ASN.1 character string types are:

- **NumericString:** strings consisting of digits (0–9) and spaces
- **PrintableString:** strings consisting of printable characters such as letters (upper and lower case), digits, punctuation marks and spaces
- **IA5String:** strings consisting of characters from the CCITT International Alphabet 5 which is essentially the ASCII character set
- **TeletexString (T61String):** strings of characters defined by CCITT Recommendation T.61
- **VideoString:** string of alphabetic and graphical characters defined in CCITT Recommendations T.100 and T.101

- **VisibleString (ISO646String):** string of characters from character set defined in ISO 646
- **GraphicString:** string of characters from character set defined in ISO 8824
- **GeneralString:** string of characters from character set defined in ISO 8824

Examples of some of these types declarations are:

Name ::= **PrintableString**
Tnum ::= **NumericString**
valName Name ::= "Smith"
valTnum Tnum ::= "1092620"

Tagged types

The ASN.1 standard associates a type denotation, called a *tag,* with ASN.1 types. The general form of such a tag is:

[<class> <number>]

where <class> denotes the class of the tag and <number> indicates a non-negative number associated with the tag. ASN.1 defines four classes of tags which are: universal, application, private and context-specific.

- The *universal class* tags, for some data types, are specified in the International Standard for the Abstract Syntax Notation One (ASN.1). The tag assignments for data types in the universal class are shown in Table 2.

UNIVERSAL 1	BOOLEAN
UNIVERSAL 2	INTEGER
UNIVERSAL 3	BIT STRING
UNIVERSAL 4	OCTET STRING
UNIVERSAL 5	NULL
UNIVERSAL 6	OBJECT IDENTIFIER
UNIVERSAL 7	ObjectDescriptor
UNIVERSAL 8	EXTERNAL
UNIVERSAL 9	REAL
UNIVERSAL 10	ENUMERATED
UNIVERSAL 16	SEQUENCE
UNIVERSAL 16	SEQUENCE OF
UNIVERSAL 17	SET
UNIVERSAL 17	SET OF
UNIVERSAL 18	NumericString
UNIVERSAL 19	PrintableString
UNIVERSAL 20	TeletexString (T61String)
UNIVERSAL 21	VideotexString
UNIVERSAL 22	IA5String
UNIVERSAL 23	UTCTime
UNIVERSAL 24	GeneralizedTime
UNIVERSAL 25	GraphicString
UNIVERSAL 26	VisibleString (ISO646String)
UNIVERSAL 27	GeneralString

Table 2: Universal class tag assignments

- The *application class* tags are assigned by other protocol standards. An application tag number cannot be used more than once within an ASN.1 module.
- The *private class* tags are enterprise specific and are not assigned by any International standard. A private tag number should not be used more than once within the same enterprise.
- The *context-specific* class tags have no restriction on their assignments. Their interpretation follows from the context of their use.

continued on next page

Abstract Syntax Notation One *(continued)*

A data type with an attached tag defines a new type. This new tagged type is isomorphic with the base type but has a different tag. If the keyword **IMPLICIT** appears in the definition of a tagged type, it indicates that the new tag should override an existing tag associated with the base type. A new tagged type can be defined as:

TaggedType ::= <tag> BaseType

Examples of some tagged type declarations for a personnel record structure [1] and a value declaration for John Smith's record are:

```
-- description of personnel record structure

   PersonnelRecord ::= [APPLICATION 0] IMPLICIT SET {
                 Name,
                 title [0] IA5String,
                 EmployeeNumber,
                 dateOfHire [1] Date,
                 nameOfSpouse [2] Name,
                 [3] IMPLICIT SEQUENCE OF ChildInformation DEFAULT { } }
   ChildInformation ::= SET {
                 Name
                 dateOfBirth [0] Date}
   Name ::= [APPLICATION 1] IMPLICIT SEQUENCE {
                 givenName IA5String,
                 initial IA5String,
                 familyName IA5String }
   EmployeeNumber ::= [APPLICATION 2] IMPLICIT INTEGER
   Date ::= [APPLICATION 3] IMPLICIT IA5String-- YYYYMMDD

-- description of John Smith's personnel record value

{
   {givenName "John", initial "P", familyName "Smith" },
   title "Director",
   51,
   dateOfHire "19710917"
   nameOfSpouse {givenName "Mary", initial "T", familyName "Smith" },
     {
             {
                       {givenName "Ralph", initial "T", familyName "Smith" },
                       "19571111" },
             {
                       {givenName "Susan", initial "B", familyName "Jones" },
                       "19590717" } } }
```

Other useful types

ASN.1 defines several types or type notations which are useful in a number of applications. These types are:

- **GeneralizedTime:** a type defined in terms of other ASN.1 type such as **VisibleString**, but with some restrictions on the values of the type being used
- **UTCTime:** a type defined in terms of other ASN.1 type such as **VisibleString**, but with some restrictions on the values the type being used
- **EXTERNAL:** a type that is defined in some external document, it need not be one of the valid ASN.1 types
- **ANY:** the union of all types
- **NULL:** the empty type; a complement of type **ANY**. This type has one distinguished value which is also denoted by NULL

Examples of some of these types and values declarations are:

```
MessageContent ::= ANY            -- Any type
FileContents ::= EXTERNAL         -- External type
GeneralizedTime ::= [UNIVERSAL 24] IMPLICIT VisibleString
UTCTime ::= [UNIVERSAL 23] IMPLICIT VisibleString
eventTime GeneralizedTime ::= "19851106210627.3"
                                  -- local time 6 minutes, 27.3 seconds after
                                  -- 9 pm on November 6, 1985
```

Subtypes

The ASN.1 *subtype* is a data type refinement mechanism. The type that is being refined is called the parent type. Six subtype specifications are possible which are: single value subtype, contained subtype, permitted alphabet subtype, value range subtype, size range subtype, and inner subtype. Some examples of subtype specifications are:

```
TouchTone ::= IA5String ("1"|"2"|"3"|"4"|"5"|"6"|"7"|"8"|"9"|"0"|"*"|"#")
DaysOfWeek ::= ENUMERATED (sun(1),mon(2),tue(3),wed(4),thu(5),fri(6),sat(7))
Weekend ::= DaysOfWeek (sat|sun)
LongWeekEnd ::= DaysOfWeek (INCLUDES Weekend|mon)
ParentType ::= INTEGER
NegativeNumbers ::= ParentType (MIN..<0)
```

Macros

The ASN.1 *macro* facility is a powerful feature which allows one to create and use extended notation for types and values. Unlike the macro facilities in most other languages which do textual substitutions, ASN.1 macro facility literally extends ASN.1 grammar which makes it difficult to generate efficient compilers for ASN.1. The general structure of a macro definition is:

```
<macro-name>        MACRO ::=
                    BEGIN
                        TYPE NOTATION ::= <type-syntax>
                        VALUE NOTATION ::= <value-syntax>
                        <supporting-syntax>
                    END
```

where <macro-name> stands for the macro name. The <type-syntax> and <value-syntax> define new notations for type and value, and <supporting-syntax> provides additional grammar rules for the macro's types and values.

The key idea underlying a macro definition is the definitions of two grammars, one for type and the other for value notation. These grammars consists of one or more production rules with alternatives. Each alternative consists of a series of syntax items. Possible values for a syntax item include literal string (quoted), production reference, symbol type, symbol value(...). symbol for string, symbol for ASN.1 identifier, and ASN.1 number. An example of a macro definition is:

```
OBJECT-TYPE MACRO ::=
BEGIN
        TYPE NOTATION ::= "SYNTAX" type (TYPE ObjectSyntax)
                          "ACCESS" Access
                          "STATUS" Status
        VALUE NOTATION ::= value (VALUE ObjectName)
        Access ::= "read-only"
                 | "read-write"
                 | "write-only"
                 | "not-accessible"
        Status ::= "mandatory"
                 | "optional"
                 | "obsolete"
END
```

continued on next page

Abstract Syntax Notation One *(continued)*

The invocation of this macro returns a value of type ObjectName (which is defined in terms of **OBJECT IDENTIFIER** in Appendix 1). An instance, named sysDescr, of this macro is created as:

```
sysDescr        OBJECT-TYPE
                SYNTAX OCTET STRING
                ACCESS read-only
                STATUS mandatory
                ::= { system 1 }
```

where OCTET STRING following SYNTAX is coming from the definition of ObjectSyntax as a choice type.

Modules

ASN.1 provides a *module* concept which is similar to the module concept in other programming languages such as Modula. An ASN.1 module has a name, body, import list and export list. The general structure of a module definition is:

```
<module-name> <object-identifier> DEFINITIONS ::=
                BEGIN
                        EXPORT
                                <export-objects-list>
                        IMPORT
                                <import-objects-list>
                        <module-body>
                END
```

where <module-name> and <object-identifier> stand for the human-readable name and authoritative name (optional) of the module respectively. The <export-objects-list> defines a list of objects which this module is making available to other modules. Similarly, the <import-objects-list> defines a list of objects (along with module names using **FROM** construct) which this module is importing from other modules. For an object to be imported into a module, it must be exported by some other module. The <module-body> includes declarations for ASN.1 types, values and macros.

An example of an ASN.1 module is given in Appendix 1. This module defines the *Structure of Management Information* (SMI) for network management of TCP/IP-based internets [13–15]. The module has name RFC1155-SMI. The module is exporting all its objects and importing none. A set of **OBJECT IDENTIFIER** instances define the naming tree. The module contains a macro named OBJECT-TYPE. The remaining parts of this module are self-explanatory and are commented on in the specification. *[Ed.: RFC 1155 has since been obsoleted by RFC 1212, the modules used in this article are for illustrative purposes only.]*

Encoding rules

In addition to abstract syntax for describing data types, a procedure is needed for encoding and decoding data values for transfer from a sender to a receiver across the network. The ISO Standard 8825, Basic Encoding Rules (BER) for Abstract Syntax Notation One (ASN.1), describes the transfer syntax for encoding data to be transmitted. The BER encoding procedure is based on a *type-length-value-end* (TLVE) encoding scheme where V and E parts may be absent.

Encoding in general follows a four part structure:

- T: Type or Identifier Octets
- L: Length Octets
- V: Value or Contents Octets
- E: End-of-Contents Octets

Figure 2 shows two possible forms of encoding—definite and indefinite forms.

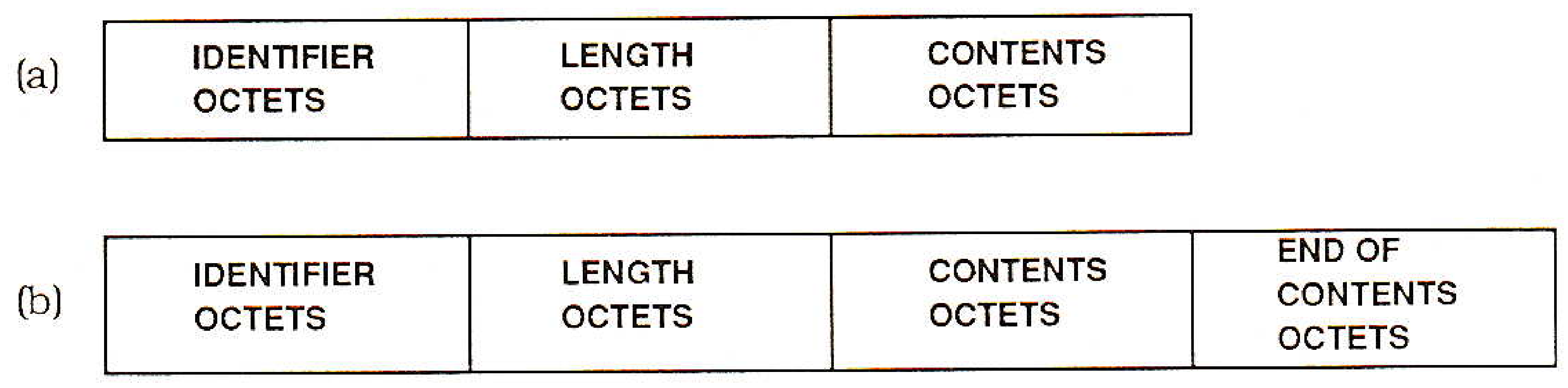

Figure 2: General structure of encoding:
(a) definite form, (b) indefinite form

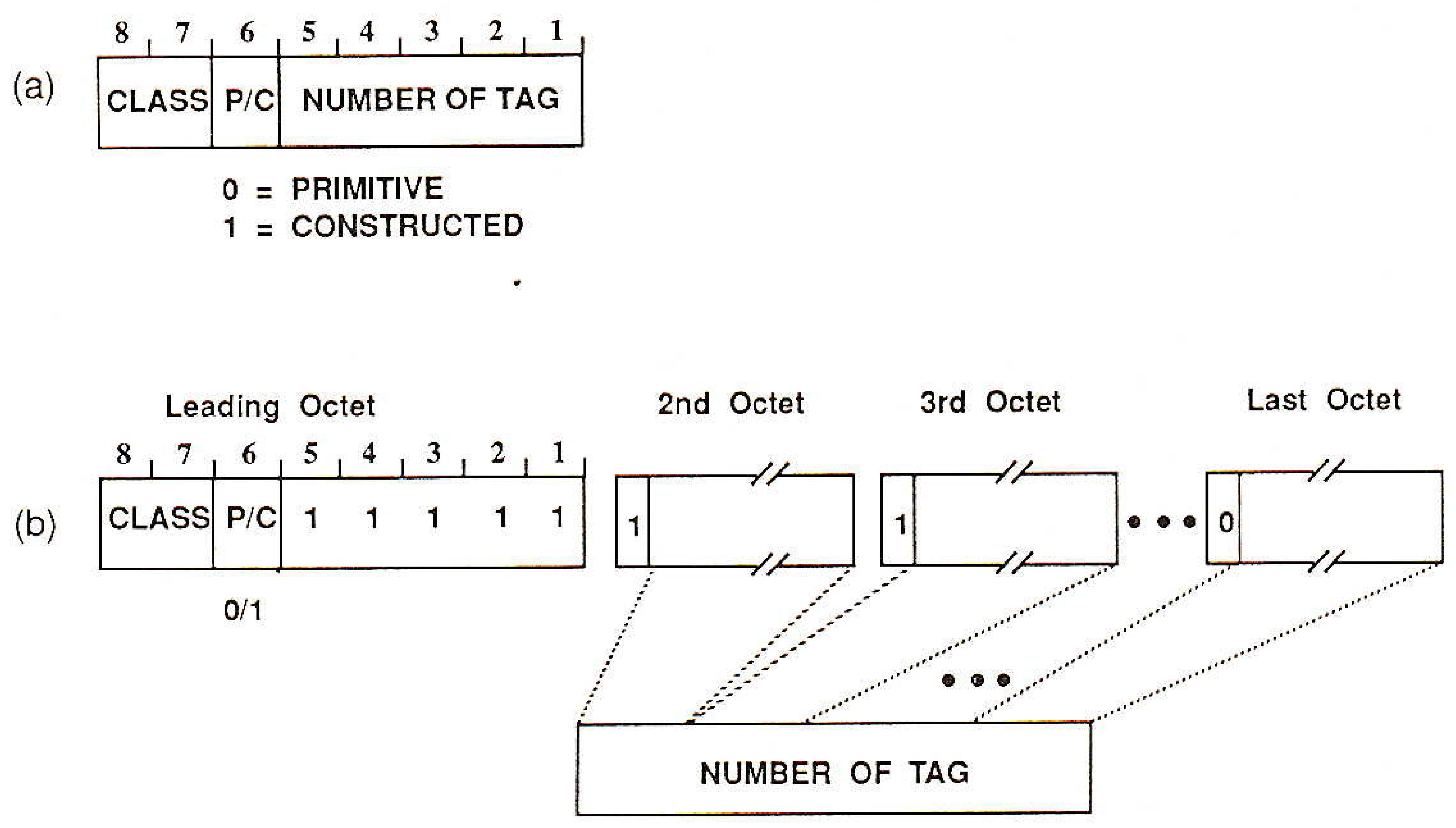

Figure 3: Identifier octets:
(a) low tag number (≤ 30), (b) high tag number (≥ 31)

Encoding the type

The encoding of the identifier (or type) octets field is shown in Figure 3. It contains the tag (class, number) for the type of the data value carried in the contents octets and also a P/C bit. The P/C bit is set to 1(0) if the data type is primitive (constructed). If the tag number of the data type is less than or equal to 30, then there is a single identifier octet as shown in Figure 3(a). For data types with tag number greater than 30, the identifier field consists of several octets as shown in Figure 3(b) in which bits 1 to 5 of the leading octet and bit 8 of each succeeding octet are set to 1 and bit 8 of the last octet is set to 0. The tag number in this case consists of the concatenation of bits 1 to 7 of all octets as indicated.

The encoding of two bits (bit 7 and 8) in the class field of the identifier octets are shown in Figure 4 for different classes of tags.

Class	Bit 8	Bit 7
Universal	0	0
Application	0	1
Context-Specific	1	0
Private	1	1

Figure 4: Encoding of class field of tag

continued on next page

Abstract Syntax Notation One *(continued)*

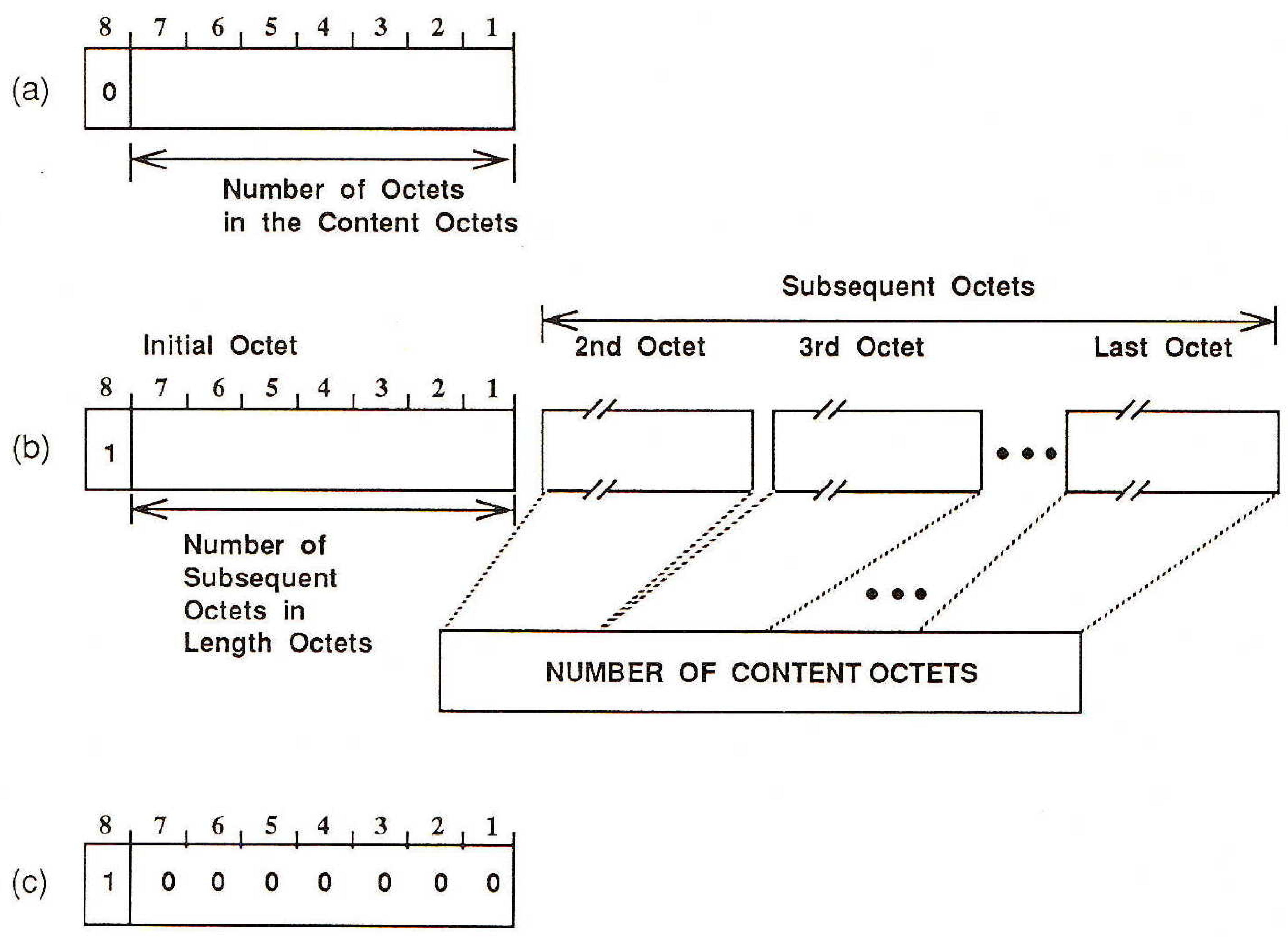

Figure 5: Encoding of length octets:
(a) short definite form, (b) long definite form, (c) indefinite form

Encoding the length

Three different types of encodings are used for the length field as shown in Figure 5. For short definite form there is a single length octet with bit 8 set to 0. The value in bits 1 to 7 gives the number of octets in the content octets as shown in Figure 5 (a). For long definite form, the length octets consist of several octets with bit 8 set to 1 in the leading octet as shown in Figure 5 (b). The value in bits 1 to 7 of the leading octet is interpreted to give the number of subsequent octets in the length octets. The number of octets in content octets is given by the concatenation of the bits in the second to the last octets of the length octets. In indefinite form of encoding, there is a single length octet with all bits set to 0, except bit 8 which is set to 1 as shown in figure 5 (c). This form is used only when contents octets are followed by end-of-contents octets.

Encoding the value

The contents octets (value) field contains zero or more octets of data values.

The end-of-contents octets field is present only if the indefinite form of encoding is used for the length octets. It consists of two zero octets as shown in Figure 6.

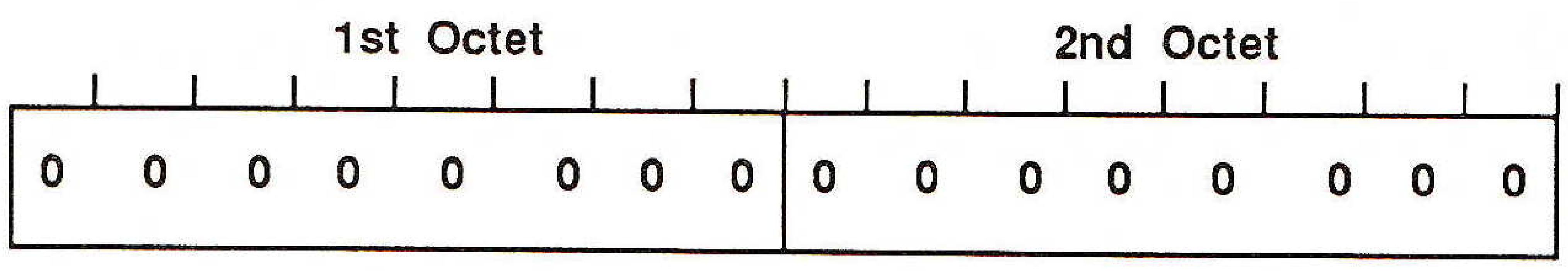

Figure 6: Encoding of end-of-contents octets

Encoding examples

We now discuss some examples of the applications of the BER of ASN.1 for encoding values of some data types.

- **BOOLEAN:** A value of this data type is encoded in primitive form with tag of universal 1 (see Table 1). A value of FALSE (TRUE) is encoded as zero (non-zero) value in the contents octets. Therefore, it requires one octet for length octets and one octet for contents octets. The value TRUE can be encoded as:

Boolean	Length	Contents
01_{16}	01_{16}	FF_{16}

- **INTEGER:** A value of this data type is encoded in primitive form with tag of universal 2. An integer value is encoded as one or more contents octets. The bits in the first contents octets shall not be all 1's or all 0's. The contents octets contain a two's-complement binary number equal to integer value. The binary number is obtained by concatenating bits of all contents octets starting with the first octet, followed by the second octet and so on. The decimal value of 43 can be encoded as:

Integer	Length	Contents
02_{16}	01_{16}	$2B_{16}$

- **BITSTRING:** A value of this data type is encoded either in primitive or constructed form with tag of universal 3. A **BITSTRING** value is encoded as one or more contents octets.

  In primitive encoding, the contents octets contain the initial octet followed by zero or more subsequent octets. The bits of the bit string are placed in the subsequent octets, starting with the bit 8 of first subsequent octet. The last subsequent octet may contain some unused bits. The count of the unused bits in the last subsequent octet is put in the initial octet as an unsigned binary integer with bit 1 as the least significant bit.

  In constructed encoding, the contents octets contain complete encoding of zero or more data values (of type **BITSTRING**). Each such encoding includes identifier, length and contents octets (may include end-of-contents octets). The encoding of each data value can be either primitive or constructed. Two possible encodings of **BITSTRING** value of '0A3B5F291CD'H are:

Primitive Encoding

BitString	Length	Contents
03_{16}	07_{16}	$040A3B5F291CD0_{16}$

Constructed Encoding

BitString	Length	Contents		
23_{16}	08_{16}			
		BitString	Length	Contents
		03_{16}	03_{16}	$000A3B_{16}$
		03_{16}	05_{16}	$045F291CD0_{16}$
		EOC	Length	
		00_{16}	00_{16}	

continued on next page

Abstract Syntax Notation One *(continued)*

- **SEQUENCE:** A value of this type is encoded in constructed form. The content octets consist of complete encoding of data values of each data type in the sequence type in the order in which these types appear in the sequence type.

  If a component type in sequence type is labeled with keyword **OPTIONAL**, the corresponding data value may be absent. However, if a data value is present for a type labeled with keyword **OPTIONAL** or **DEFAULT**, it appears in its proper place in the order of types in sequence type. A value { "Smith" **IA5String** , ok **BOOLEAN** } of sequence type can be encoded as:

Sequence	Length	Contents		
30_{16}	$0A_{16}$			
		IA5String	Length	Contents
		16_{16}	05_{16}	"Smith"
		Boolean	Length	Contents
		01_{16}	01_{16}	FF_{16}

- **CHOICE:** An encoded value of this type is the same as the encoding of the chosen type.

- **Tagged Type:** The encoding of a tagged value is derived from the complete encoding of the base type in the tagged type.

  The encoded value depends on the presence or absence of keyword **IMPLICIT** in the definition of tagged type. If this keyword is used, then the encoding is constructed and the contents octets consist of complete base encoding. However, if this keyword is not present, then (1) the encoding is constructed if the base encoding is constructed and primitive otherwise, (2) the content octets are the same as the content octets for the base encoding.

- **OBJECT IDENTIFIER:** A value of this type is encoded in primitive form. The content octets is an ordered list of encoding of the subidentifiers concatenated together. The number of subidentifiers is one less than the number of components in the object identifier value.

  The first subidentifier is given by $(X*40) + Y$ where X and Y are the first and second components of the object identifier value. The numerical value of the i-th ($i > 1$) subidentifier is equal to the numerical value of the (i+1)-th component in the object identifier value. Each subidentifier is represented as a series of one or more octets. Bit 8 of each octet is set to 1 except for the last octet which is set to 0. An **OBJECT IDENTIFIER** value of:

  { 2 100 3 }

  has first and second subidentifier equal to 180 and 3 respectively. The encoding of this **OBJECT IDENTIFIER** value is:

OBJECT IDENTIFIER	Length	Contents
06_{16}	03_{16}	$FF3503_{16}$

The personnel record example discussed earlier contains several data types. The encoding of a value for John Smith's record is shown in Figure 7 [1].

```
Personnel
Record Length Contents
60     8185
          Name Length Contents
          61    10
                   VisibleString Length Contents
                   1A            04     "John"
                   VisibleString Length Contents
                   1A            01     "P"
                   VisibleString Length Contents
                   1A            05     "Smith"
          Title  Length Contents
          A0     0A
                   VisibleString Length Contents
                   1A            08     "Director"
          Employee Number Length Contents
          42              01     33
          Date of Hire    Length Contents
          A1              0A
                               Date Length Contents
                               43   08     "19710917"
          Name of Spouse  Length Contents
          A2              12
                               NameLength Contents
                               61    10
                                        VisibleString Length Contents
                                        1A            04     "Mary"
                                        VisibleString Length Contents
                                        1A            01     "T"
                                        VisibleString Length Contents
                                        1A            05     "Smith"
          [3]             Length Contents
          A3              42
                               Set  Length Contents
                               31   1F
                                        Name Length Contents
                                        61    11
                                                  VisibleString Length Contents
                                                  1A            05     "Ralph"
                                                  VisibleString Length Contents
                                                  1A            01     "T"
                                                  VisibleString Length Contents
                                                  1A            05     "Smith"
                                        Date of
                                        birth Length Contents
                                        A0    0A
                                                  Date Length Contents
                                                  43   08     "19571111"
                               Set  Length Contents
                               31   1F
                                        Name Length Contents
                                        61    11
                                                  VisibleString Length Contents
                                                  1A            05     "Susan"
                                                  VisibleString Length Contents
                                                  1A            01     "B"
                                                  VisibleString Length Contents
                                                  1A            05     "Jones"
                                        Date of
                                        birth Length Contents
                                        A0    0A
                                                  Date Length Contents
                                                  43   08     "19590717"
```

Figure 7: Encoding of Smith's personnel record value

Encoding efficiency

It has been pointed out that ASN.1 encoding, using the BER, can involve excessive overhead [11,16]. As we have seen, the ASN.1 transfer syntax is based on a type-length-value, or type-value-end format, where almost all fields have a variable size encoding. This introduces two types of overhead. First, there may be redundant type and length information, and second, the byte-based, variable sized representation consumes CPU time in encoding and decoding. In a simple experiment, [16] found that the ASN.1 conversions took between 5 and 20 times the time for a simple memory copy. This is a major concern, as it is likely that high speed networks will be limited by speed of the connected systems. Another concern is that array length is not available until end of array is received, which may cause storage management problems for the receiver.

continued on next page

Abstract Syntax Notation One *(continued)*

It is desirable to develop improved ASN.1 encoding rules which reduce the amount of overhead. In [16], "light-weight encoding principles" are proposed, which abandon systematic TLVE coding. These principles are: first, avoid unnecessary information, i.e., redundant or implicit type and length information, and second, use a fixed representation whenever possible. In experiments comparing such a "lightweight encoding" with ASN.1, they found that overall resulting message length was similar, and coding and decoding time was up to 6 times faster. They proposed a possible solution to this problem—an ASN.1 format negotiating phase resulting in a choice of either (1) continued ASN.1 communication, (2) compressed encoding, for low speed nets, or (3) "light weight encoding" for high speed nets.

A scheme similar to the lightweight encoding of [16] is used by INMOS for message passing between *transputers* (microprocessors designed for distributed processing) [19]. In *occam* (the transputer programming language) data types are defined as *channel protocols,* and are a subset of the data types in ASN.1 [20]. Static type checking is used to guarantee that type tags and length fields are used only when they are actually needed. Rather than arbitrarily extensible real and integer types, *occam* provides a fixed set: INT16, INT32, INT64, REAL32, and REAL64, with the real numbers in IEEE format. There are two sorts of varying length type structures. In the array protocol, the array is preceded by either a byte or integer length field; the choice of integer or byte is declared by the user. In the "variant protocol" the message type is a user defined tag, which is encoded in a single byte. Simple, array, and variant types can be combined to form more complex structures. The result is a combination of much of the descriptive power of ASN.1 with a highly efficient encoding.

ASN.1 tools

ASN.1 is a formal notation for describing data types and encoding data values, and working directly with ASN.1 can be quite tedious. Automated tools can be developed for (1) creating ASN.1 descriptions, (2) pretty printing of ASN.1 descriptions, (3) syntax checking of ASN.1 descriptions, (4) translating ASN.1 types into types of a programming language, and (5) creating encoding and decoding routines for data types. Some examples of such tools are discussed in [10,11]. Some of the difficulties involved with creating such tools include the problem of generating efficient (as opposed to simply correct) code for encoding and decoding data types, and the problem of implementing the ASN.1 macro facility in its full generality [11].

Summary and conclusions

ASN.1 provides a powerful set of features for unambiguous and machine-independent definitions of data types. It also provides encoding rules for unambiguous representation of data values for transmission across the network. ASN.1 has been used successfully in several protocols at the application level and also in protocols for network management.

ASN.1 is not entirely satisfactory as a language for abstract data types. It lacks a formal semantics, and does not integrate nicely with FDTs such as *Estelle* and *LOTOS* [17,18]. Also, as we have noted, the encoding rules of ASN.1 may involve excessive overhead. These difficulties may be acceptable at present, in light of the strong need for a standard representation for communication of abstract data types. We expect that ongoing work within ISO on improvement of encoding rules to at least partially rectify the encoding efficiency problem.

References

[1] ISO, Information Processing Systems—Open Systems Interconnection—Specification of Abstract Syntax Notation One (ASN.1), ISO International Standard 8824, 1987.

[2] ISO, Information Processing Systems—Open Systems Interconnection—Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1), ISO 8825, 1987.

[3] CCITT, Specification of Abstract Syntax Notation One (ASN.1), CCITT Recommendation X.208, 1988.

[4] CCITT, Specification of Basic Encoding Rules for ASN.1, CCITT Recommendation X.209, 1988.

[5] ISO, Information Processing Systems—Open Systems Interconnection—Abstract Syntax Notation One (ASN.1), Draft Addendum 1: Extensions to ASN.1. International Organization for Standardization and International Electrotechnical Committee, 1987. Draft Addendum 8824/DAD 1.

[6] ISO, Information Processing Systems—Open Systems Interconnection—Abstract Syntax Notation One (ASN.1)—Draft Addendum 1: Extensions to ASN.1 Basic Encoding Rules. ISO and IEC, 1987. Draft Addendum 8825/DAD 1.

[7] D. Chappell, "A Tutorial on Abstract Syntax Notation One (ASN.1)," Omnicom Information Service, Report #25, Omnicom, Inc., VA, December 1986.

[8] P. Gaudette, "A Tutorial on ASN.1," NIST Technical Report NCSL/SNA-89/12, May 1989.

[9] G. Neufeld & S. Vuong, "An Overview of ASN.1," UBC Technical Report, 1990.

[10] G. Neufeld & Y. Yang, "An ASN.1 to **C** Compiler," *IEEE Transactions in Software Engineering,* 1990.

[11] M. T. Rose, *The Open Book: A Practical Perspective OSI,* Prentice-Hall, 1989.

[12] D. P. Sidhu, *OSI Conformance Testing,* Prentice-Hall, Englewood Cliffs, New Jersey, (to be published in 1992).

[13] M. Rose & K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based Internets," RFC 1155.

[14] K. McCloghrie & M. Rose, "Management Information Base for Network Management of TCP/IP-based Internets," RFC 1156.

[15] J. Case, M. Fedor, M. Schoffstall & J. Davin, "A Simple Network Management Protocol (SNMP)," RFC 1157, May 1990.

[16] C. Huitema & A. Doghri, "Defining Faster Transfer Syntaxes for the OSI Presentation Protocol," *Computer Communications Review,* Vol. 19, No. 5, October 1989.

[17] G. v. Bochmann & M. Deslauriers, "Combining ASN.1 Support with the LOTOS language," Proc. *IFIP Symposium on Protocol Specification, Testing and Verification XI,* North Holland, June 1989.

[18] G. v. Bochmann, D. Ouimet & G. Neufeld, "Implementation Support Tools for OSI Application Layer Protocols," UBC Technical Report, 1990.

[19] INMOS Corp., *Communicating Process Architecture,* Prentice Hall 1988.

continued on next page

Abstract Syntax Notation One *(continued)*

[20] INMOS Corp., *occam 2 Reference Manual,* Prentice-Hall, 1988.

[21] M. T. Rose, *The Simple Book: An Introduction to Management of TCP/IP-based internets,* Prentice-Hall, 1990.

HOWARD MOTTELER received his B.S. (1980) from University of Puget Sound, his M.S. from Purdue (1982) and his Ph.D. from University of Maryland at College Park, (1987). He has been an assistant professor at UMBC for the last 4 years. Research interests center on distributed systems, including process theory, task assignment, and conformance testing. He is an active member of transputer user's groups, and has developed a transputer-based course in distributed programming for undergraduates at UMBC. Current projects include developing a graphic environment for distributed programming.

DEEPINDER SIDHU received his B.S. degree in Electrical Engineering from the University of Kansas, and the M.S. and Ph.D. degrees in Computer Science and Theoretical Physics respectively from the State University of New York, Stony Brook. He worked at Rutgers University, Brookhaven National Laboratory, Mitre Corp., SDC-Burroughs Corp. (now Unisys), and Iowa State University. He is currently Professor of Computer Science with the University of Maryland Baltimore County (UMBC) campus and the University of Maryland Institute for Advanced Computer Studies (UMIACS) at College Park. He has published over 100 papers in Theoretical Physics and Computer Science. His current research interests are in the areas of computer networks and distributed systems. He is the Editor-in-Chief of *Journal of High Speed Networks,* a new international journal. He is the author of a graduate level text, *OSI Conformance Testing,* to be published in 1992.

Appendix: ASN.1 Module Specification

This appendix gives an example of an ASN.1 module. This module defines the *Structure of Management Information* (SMI) for network management of TCP/IP-based internets [13–15]. The module has name RFC1155-SMI. It is exporting all its objects and importing none. The module contains a macro named OBJECT-TYPE.

```
RFC1155-SMI DEFINITIONS ::= BEGIN

    EXPORTS -- EVERYTHING
      internet, directory, mgmt,
      experimental, private, enterprises
      OBJECT-TYPE, ObjectName, ObjectSyntax, SimpleSyntax,
      ApplicationSyntax, NetworkAddress, IpAddress,
      Counter, Gauge, TimeTicks, Opaque;

    -- the path to the root

    internet OBJECT IDENTIFIER ::= { iso org(3) dod(6) 1 }
    directory OBJECT IDENTIFIER ::= { internet 1 }
    mgmt OBJECT IDENTIFIER ::= { internet 2 }
    experimental OBJECT IDENTIFIER ::= { internet 3 }
    private OBJECT IDENTIFIER ::= { internet 4 }
    enterprises OBJECT IDENTIFIER ::= { private 1 }

    -- definition of object types

    OBJECT-TYPE MACRO ::=
    BEGIN
            TYPE NOTATION ::= "SYNTAX" type (TYPE ObjectSyntax)
                              "ACCESS" Access
                              "STATUS" Status
            VALUE NOTATION ::= value (VALUE ObjectName)
            Access ::= "read-only"
                     | "read-write"
                     | "write-only"
                     | "not-accessible"
            Status ::= "mandatory"
                     | "optional"
                     | "obsolete"
    END
```

```
-- names of objects in the MIB
ObjectName ::= OBJECT IDENTIFIER
-- syntax of objects in the MIB
ObjectSyntax ::=
        CHOICE {
                simple
                        SimpleSyntax,
-- note that simple SEQUENCEs are not directly mentioned here to
-- keep things simple (i. e., prevent mis-use). However, application-wide
-- types which are IMPLICITly encoded simple SEQUENCEs may
-- appear in the following CHOICE
                application-wide
                        ApplicationSyntax
        }
SimpleSyntax ::=
        CHOICE {
                number
                        INTEGER,
                string
                        OCTET STRING,
                object
                        OBJECT IDENTIFIER,
                empty
                        NULL
        }
ApplicationSyntax ::=
        CHOICE {
                address
                        NetworkAddress,
                counter
                        Counter,
                gauge
                        Gauge,
                ticks
                        TimeTicks,
                arbitrary
                        Opaque
-- other application-wide types, as they are defined, will be added here
        }
-- application-wide types
NetworkAddress ::=
  CHOICE {
        internet
                IpAddress
        }
IpAddress ::=
  [APPLICATION 0]                      -- in network-byte order
        IMPLICIT OCTET STRING (SIZE (4))
Counter ::=
  [APPLICATION 1]
        IMPLICIT INTEGER (0..4294967295)
Gauge ::=
  [APPLICATION 2]
        IMPLICIT INTEGER (0..4294967295)
TimeTicks ::=
  [APPLICATION 3]
        IMPLICIT INTEGER (0..4294967295)
Opaque ::=
  [APPLICATION 4]                      -- arbitrary ASN.1 value,
        IMPLICIT OCTET STRING -- "double-wrapped"
END
```

Réseaux Associés pour la Recherche Européenne (RARE)

Networking for Researchers in Europe

Introduction

Réseaux Associés pour la Recherche Européenne (RARE) is the association of European networking organizations and their users. The aim of RARE is to foster cooperation between both national and international networking organizations in order to develop a harmonized data communications infrastructure. Such an infrastructure will enable researchers to communicate, use information and access computer resources throughout Europe and in other continents. To achieve this aim, RARE supports the principles of open systems as defined by the International Organization for Standardization and the work on functional profiles undertaken by the *European Workshop for Open Systems* (EWOS) and the *European Telecommunications Standards Institute* (ETSI).

In 1985 a workshop was held in Luxembourg during which the idea of establishing an association to foster European research networking was born. In the five years since RARE's formal founding under Dutch law in 1986, the Association has grown to a membership representing more than 20 countries, plus nine international organizations supporting research in Europe.

Since its foundation, RARE has made a number of significant contributions to the advancement of international networking for researchers in Europe and is accepted by governments as the representative body for research networking. Through its Working Groups RARE brings together European technical experts and through them it has a voice on all of the major international bodies concerned with open communications and standardization.

Members

The Association has four types of members. The twenty European countries listed in the statutes are eligible to be the National Members which, as voting members, formally constitute the Association. Associate National Members are national research networking organizations in other countries which support the objectives of RARE and are closely associated with the use, coordination and provision of an infrastructure to the benefit of the research community. Liaison Members are organizations which are involved in networking and related matters with whom RARE considers it important to have close and continuing contact. As of July 1991, the membership of RARE comprised the following organizations and representatives.

Full National Members:

Austria	Iceland	Spain
Belgium	Ireland	Sweden
Denmark	Italy	Switzerland
Finland	Luxembourg	Turkey
France	Netherlands	United Kingdom
Germany	Norway	Yugoslavia
Greece	Portugal	

The *Commission of the European Communities* (CEC) DG XIII participates actively in RARE's work in view of its special responsibilities for information technology and the infrastructure for research in Europe.

Associate National Members:

Czechoslovakia	Israel	Soviet Union
Hungary	Poland	
India	Republic of Korea	

International Members:

CERN	ECFA	ESA	EurOpen	NORDUNET
EARN	ECMWF	ESONE	JINR	

Liaison Members:

CREN (BITNET and CSNET)

Membership plans for the 1990s

It has been the continuing policy of the *RARE Council of Administration,* the governing body of the Association, to encourage the growth of RARE's membership to encompass national research networking organizations in countries where the objectives of RARE are supported. In the short term this policy is expected to result in participation of all European countries and their networking organizations. RARE's statutes are currently under review in order to accommodate the evolving political situation in Europe.

Working Groups

The technical backbone of RARE is formed by the RARE *Working Groups* which are responsible for developing coordination and cooperation in technical areas. These Working Groups bring together some of the foremost experts in open systems, standards and networking operations in Europe today. Membership of the Working Groups is by nomination from the national networking organizations, together with a number of invited experts. The subject areas of the Working Groups are as follows:

Working Group 1: Message Handling Systems

- Create and promote a European infrastructure for a message handling service within the European research community, with connections to the global environment.

Working Group 2: File Transfer, Access and Management

- Prepare for an efficient transition to FTAM
- Monitor the functions/features of the base standards and profiles
- Define and conduct interoperability tests.

Working Group 3: Directory and User Information Services

- Promote the establishment of information and user support services in the RARE countries as well as directory services based on the X.500 standard.

Working Group 4: Network Operations and X.25

- Coordinate the exchange of management information between network service providers in the RARE community.
- Address the related technical issues.

Working Group 5: Full Screen Services

- Promote the use of OSI conformant Virtual Terminal Protocols
- Conduct pilot activities
- Track the related technical developments.

Working Group 6: High Speed Communications and ISDN

- Address the technical and management problems related to high speed networking between the RARE countries.

Working Group 8: Management of Network Application Services

- Address the problems of accounting, charging and costs distribution, and security and authentication.

continued on next page

RARE *(continued)*

Other working bodies

Key elements also supporting the aims of RARE are:

- The adoption of RIPE (*Réseaux IP Européens*)—a forum for the exchange of technical information and the creation of expertise on Internet Protocol networking in Europe
- The setting up of the EEPG (European Engineering and Planning Group)—chartered with the study, preparation and planning of a possible European data communications backbone.

MHS Pilot Project

In 1987 RARE, with funding from the CEC, France and Norway undertook a pilot project aimed at the creation of a broadly based pilot infrastructure of emerging electronic mail implementations of the X.400 standard. The project was executed over a two year period by the University of Trondheim Computer Centre, RUNIT, in Norway and resulted in more than 100,000 users connected through 530 computer installations in 20 countries using 15 different X.400 implementations, the public X.25 network being used for all international traffic. The work of the pilot project has led directly to the creation of COSINE Service S2: Message Handling Services.

EUREKA COSINE project

During 1987 and 1988 RARE, supported by its Working Groups, undertook a number of studies that led to a series of documents: the *COSINE Specification Phase* reports. These documents reviewed the state of research networking in Europe, the availability of standards and made recommendations for a future work programme.

During 1990 the *COSINE Implementation Phase Execution Contract* (CIPEC) was signed between RARE and the CEC on behalf of COSINE. Under this contract, RARE has established, and is operating, a *COSINE Project Management Unit* (CPMU) to perform the individual tasks and manage the sub-projects and pilot services that are part of the Project, the final objective of COSINE being the establishment of a self-sustaining networking infrastructure for industrial and academic research workers from 1993 onwards. Projects and Services currently in operation are:

- P1.1 Pilot Gateway Services to the USA — FTAM
- P2.1 Pilot Information Services — *Pilot International Directory Services* (PARADISE)
- P2.2 Pilot Information Services — *Pilot Support and Information Services* (CONCISE)
- P3 Support of International User Groups
- S1 Provision of X.25 (1984) Infrastructure (IXI)
- S2.1 Message Handling Services — Interworking of Existing X.400 Administrative Domains
- S2.2 Message Handling Services — Gateway Services to the USA

E-mail reliability

Under the tutelage of RARE, a research student produced a report evaluating the reliability of current international academic electronic messaging systems to obtain an objective idea of the quality of the service offered to researchers.

Connectionless network services pilot project

A small scale pilot project was proposed by Working Group 4 to RARE in order to gain experience with the inter-networking of products based on ISO-IP standards. The pilot project was subsequently proposed to COSINE and adopted with additional funds being allocated to supplement those being provided by the collaborators to the project.

Liaisons with other organizations

RARE provides a user's voice on a number of European standardization and political bodies, such as EEMA, EWOS, ECTUA and ETSI. On a broader scale, RARE represents the European participation on the *Coordinating Committee for Intercontinental Research Networking* (CCIRN).

Conferences

An annual *Networkshop* is organized by RARE to provide the opportunity for a broad discussion on networking for the European research community, to review progress in its area of activities and to stimulate new work. Since 1990 the workshop has been known as the *Joint Networking Conference* (JNC) which was held that year in Killarney, Ireland in association with EARN.

In this fifth anniversary year the JNC took place in Blois, France. It was organized in cooperation with EARN, EurOpen, the Internet Activities Board, NORDUNET and the French Ministry of Research and Technology.

In addition to the annual conference, RARE organizes a number of symposia related to specific technical areas. Notable amongst these have been two symposia held in 1989 and 1991 on European High Speed Networking. On both occasions the meetings were held with the support of the Commission of the European Communities DG XIII.

These conferences and symposia have clearly shown that RARE is the appropriate body to move forward with initiatives in networking technology.

For more information about RARE and its publications contact:

RARE Secretariat
Singel 466–468
NL–1017 AW Amsterdam
The Netherlands
Tel: +31 20 639 1131
Fax: +31 20 639 3289
X.400: `C=nl; ADMD=400net; PRMD=surf; O=nikhef; S=raresec`
Internet: `raresec@nikhef.nl`

References

[1] Crowcroft, J., "International Internetworking," *ConneXions,* Volume 2, No. 4, April 1988.

[2] Onions, J., "Components of OSI: The X.400 Message Handling System," *ConneXions,* Volume 3, No. 5, May 1989.

[3] Benford, S., "Components of OSI: X.500 Directory Services," *ConneXions,* Volume 3, No. 6, June 1989.

[4] Truoel, K.,"Components of OSI: File Transfer, Access, and Management (FTAM)," *ConneXions,* Volume 4, No. 4, April 1990.

[5] Vair, D., "Components of OSI: X.25—the Network, Data Link, and Physical Layers of the OSI Reference Model," *ConneXions,* Volume 4, No. 12, December 1990.

[6] Goldstein, S. & Michau, C., "Convergence of European and North American Research and Academic Networking," *ConneXions,* Volume 5, No. 4, April 1991.

[7] Stockman, B., "Current Status on Networking in Europe," *ConneXions,* Volume 5, No. 7, July 1991.

GOSIP Challenges in the Department of Defense

by Robert Cooney, US Navy

Background

The mandate for the *Government Open Systems Interconnect Profile* (GOSIP), Version 1, has been in effect for all major system procurements and upgrades since August 1990. With GOSIP Version 2 now off-the-presses, many compliant local area network (LAN) implementations have already surfaced. What seems to be missing are OSI implementations across a wide area network (WAN) such as the *Defense Data Network* (DDN). This missing link brings to the forefront the true importance of OSI, with interoperability across wide areas, even worldwide, being the common denominator. OSI makes seamless and unencumbered communication possible between multivendor computers connected by heterogeneous networks of global scope. Still GOSIP should not be viewed as just a set of worldwide networking protocols. By broadening ones view, a process begins to take shape for consolidating applications into a unified structure (i.e., architecture) that will ensure that future technology evolution remains consistent with what is currently deployed in the operational environment. Thus the intent is pretty clear: GOSIP specifications provide the detail for independent implementations of networking and application software to interoperate, independent of location, vendor specific design, or implementation technology.

Current status

The DDN, in its present state of evolution, has over 300 networks attached to it and an unknown number of hosts and users. Many of the unknowns are due to the fact that many packet switch node port connections, originally registered as host connections, now connect to gateways to other networks.

DDN has some limitations which have slowed the acceptance and use of its facilities and will further restrict the network's ability to attract new customers. Along with the fact that there is an extraordinarily lengthy waiting period, the technologies employed have become dated. There is also the high cost of connections, the network delays experienced by existing customers, lack of flexibility in accommodating new high-speed bursty applications, and the general lack of the network's capacity to expand.

There are alternatives available in the commercial market to provide much needed improvements, and recently a plan to deploy such needed technologies has been developed. The DISN (*Defense Information System Network*) pilot prototype network is being planned by the technical staff at the *Defense Information System Agency* (DISA) to explore newer and emerging communications technologies. Work is underway to prototype switched voice, video, high-speed data, cellular, and ISDN technologies. There is a need to provide more end-to-end capabilities and services but little emphasis has been given to OSI in the near term. DISA has noticed the emergence of Intelligent Circuit Multiplexers and IP Routers in Air Force and Navy subnetworks on the DDN. DISA is also prototyping on the DDN *Fast Packet Switching* which is expected to have significant economic and reliability advantages over older technologies.

OSI transition strategies

The Department of Defense (DoD) Transition strategy calls for a coexistence of both the MIL-STD TCP/IP and OSI communication protocols. Adding the ISO OSI management and administration requirements to the existing MILNET may turn out to be more than the present structure is able to handle. It must also be noted that OSI protocols call for an IP known as the *Connectionless Network Protocol* (CLNP).

But CLNP and OSI router protocols such as the *End System to Intermediate System* (ES–IS) and the *Intermediate System to Intermediate System* (IS–IS) are functional capabilities that do not exist on the current packet switches in use on the DDN today.

Long-haul networks such as the DDN are a significant concern throughout DoD. The DDN is conceivably the largest multi-vendor interoperable network in the world today, and is currently based only on the TCP/IP protocols. Its transition to OSI and to OSI-dependent hardware and software suites must be undertaken and achieved smoothly and cost effectively.

A number of initiatives are currently underway within the DoD to chart aspects of the transition process towards the use of OSI applications and communication protocols as part of an *Open System Environment,* and the DISN Pilot is a viable candidate to be one of them. Existing initiatives include the *Corporate Information Management,* the *Defense Messaging System, GOSIP Gateways,* numerous projects related to *Remote Data Base Access, Transaction Processing,* Multi-media access, Network Management, Security, *Electronic Data Interchange,* and *Computer-aided Acquisition and Logistic System* (CALS). The Navy Computer and Telecommunications Station in Washington, along with many other departments and agencies, has been involved in much of this work. However, the scope of the problem is vast. The number of installed systems within DoD utilizing OSI protocols is expected to reach into the thousands by the mid to late 1990s. These systems will include all classes of equipment: desktop PCs, workstations, LAN servers, routers, bridges, management stations, packet switching equipment, PADs, and host computers. The number of value added services needing support will multiply to encompass e-mail, voice mail, store and forward fax, videoconferencing, EDI, transaction processing, scheduling, logistics, distributed inventory resource databases, distributed personnel databases, work group management, and a host of other end-user applications requiring a heavy dose of bandwidth, multivendor interoperable messaging, and distributed database access.

Potential implementation strategies

Intelligent Circuit Multiplexers and IP Routers are possible technology candidates for improving DDN performance, controlling the cost of its communication services while also integrating voice, data, and other advanced technology services. Both of the above candidate technologies have already been deployed and implemented, and are establishing solid track records of operational performance. Fast Packet Switching appears to be an even better alternative, and all of these technologies can be made to operate with each other and with the present X.25-based DDN facilities. Adding the ISO OSI protocols would not inhibit any of the above inherent user functionality, but would significantly add depth and dimension to the interoperability issue at the DoD. A dual dynamic routing protocol capability of IS–IS and OSPF can be used as a continuing strategy for coexistence while transitioning from TCP/IP. It is not too early to begin to integrate the DoD networks. Furthermore, it may be more practical to deploy the first cut of OSI users on the DISN with its specific capabilities and new technologies rather than importing and adding new technologies and networking interoperability features into the existing DDN.

Intelligent multiplexers with their ability to interconnect without protocol preference, supporting high speed connections for voice/ data, are nicely matched with routers with their interconnect and interoperability features for supporting OSI protocols alongside TCP/IP.

continued on next page

GOSIP Challenges in the DoD *(continued)*

With the advent of Fast Packet Switching Technology, bandwidth management has taken on new meaning. This technology has been widely embraced because it provides bandwidth on demand. With this technology, and another called *Frame Relay,* it is possible to actually approach the full realization and bandwidth capacity of a T1 circuit with a rated speed of 1.544 Mbps. That speed has generally not been available because the bandwidth is normally broken into smaller channels. Since there are no channels in Fast Packet, the potential exists for getting more speed than ever before if the bandwidth is available, or merely slowed down during times of high traffic volumes.

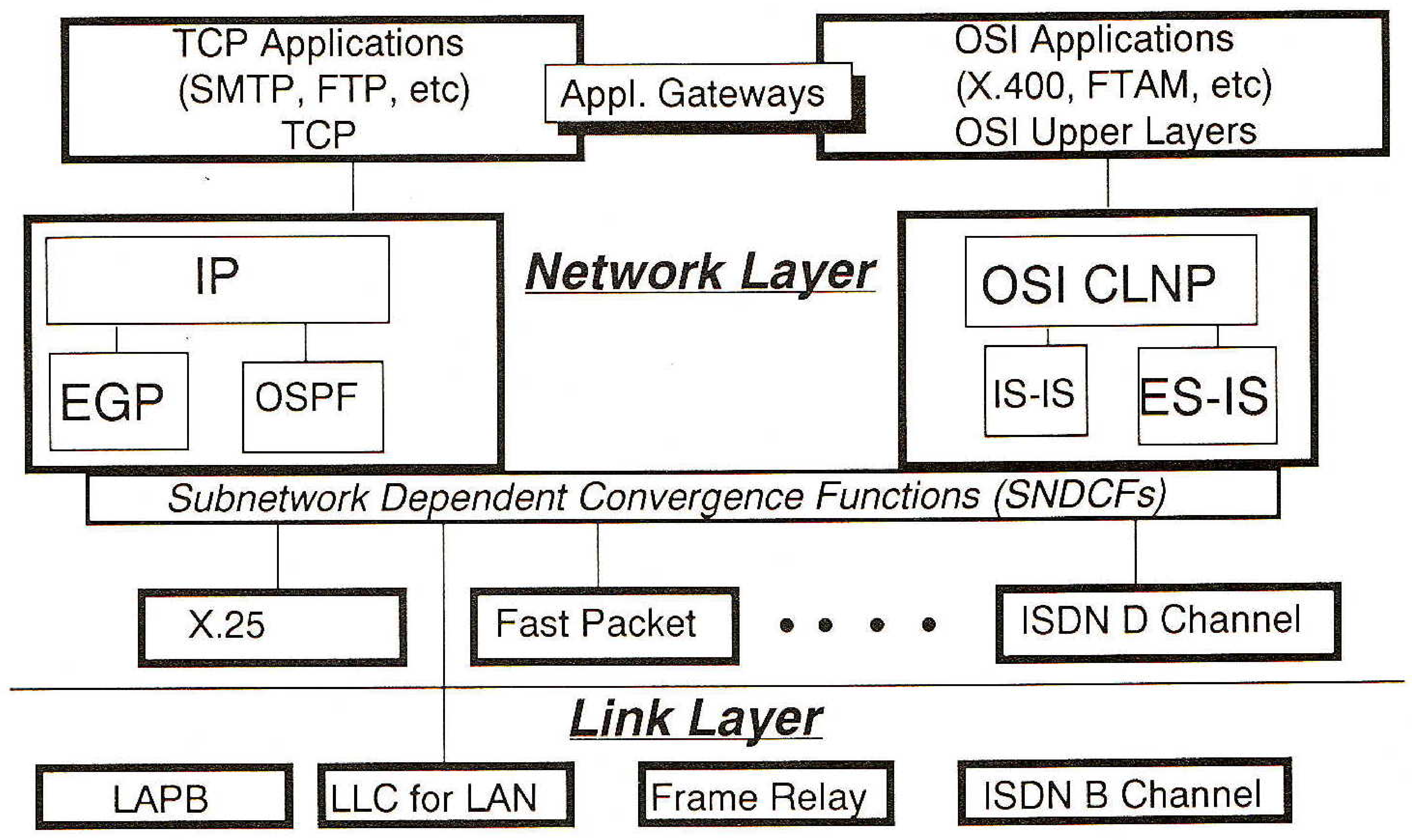

Figure 1: DISN Coexistence

Conclusion

The inclusion of the DISN prototype into the DoD OSI transitioning strategy complements other OSI transitions initiatives and also gives the GOSIP mandate of last August a higher level of credibility at the DoD. Furthermore, DoD cannot be expected to maintain its credibility with vendors if it does not itself visibly and actively promote the spread of open systems. The coexistence of both dynamic routing protocols, IS–IS and OSPF, is also a fortuitous opportunity that will solidify DoD's open system intents. In the near term, a limited deployment of OSI on the DISN will take pressure off the DDN network, while the lessons learned in a limited production environment will greatly enhance total transitioning to OSI when that goal is eventually realized. In the not-too-distant future, perhaps a stand-alone pure GOSIP based pilot/operational version of DISN (the sequel) should be considered, with a gateway to exchange traffic with the existing TCP/IP based networks.

ROBERT COONEY currently serves as the Head of the Research, Development, Test and Evaluation (RDT&E) Division for the Naval Computer and Telecommunications Station, Washington, D.C. which is the U.S. Navy's largest full service ADP support agency. Mr. Cooney has worked for the Navy for the past 18 years after receiving his B.S. degree in Mathematics from the University of Kentucky, serving two years in the U.S. Army, and receiving post-graduate certificates from the American University in Computer Systems Applications and Management Information Systems. He is credited with formation of the U.S. Navy's OSI Laboratory, recently renamed the Open Systems Environment Testbed, located at the Washington Navy Yard and is a continuing influence on the Department of Defense (DoD) plan for transitioning towards and coexistence with GOSIP. He can be reached as `cooney@wnyose.nctsw.navy.mil`.

Book Review

TCP/IP and NFS: Internetworking in a UNIX Environment by Michael Santifaller (translated by Stephen S. Wilson) Addison-Wesley, 1991 ISBN 0-201-54432-6, 235 pages.

This book appealed to me since I have used NFS, but have never dug through the internals to find all its blemishes and warts. I came away from the book, however, with some questions answered but a lot unanswered.

TCP/IP

The first half of this small book covers TCP/IP. This includes some applications (TELNET, FTP, SMTP, and TFTP), the Berkeley *r*–commands, and TCP/IP administration. Reading it reminded me of Davidson's 100-page book on TCP/IP ("An Introduction to TCP/IP," Springer-Verlag, 1988). Santifaller has some nice features, such as showing the packets (sequence numbers and flags only) exchanged during the TCP three-way handshake and connection shutdown, and showing the output of *netstat* and *trpt,* for example.

There are some flaws. Santifaller details the round-trip estimators specified by RFC 793 but only mentions Jacobson's improvements in passing. He spends almost an entire page describing *Slow Start* and congestion avoidance, but omits Jacobson's classic paper from the Bibliography. He also lists almost 40 RFCs in the Bibliography, but omits the Host Requirements RFCs.

NFS

The next half of the book talks about NFS. It starts with a description of RPC and XDR and then moves into the NFS protocols. I found the description of the NFS protocol and the MOUNT protocol more readable than RFC 1094. Chapter 10 ("Implementation of NFS") is the most interesting, and readers will probably find it the best chapter in the book. This chapter covers lots of details that aren't documented well in the various vendor's documentation: the various daemons (*biod, nfsd, statd, mountd*), restrictions (why multiple processes appending to the same file doesn't work), and diagnosing RPC and NFS problems (*rpcinfo* and *nfsstat*). I especially liked the description and picture of the locking protocol, involving the *lock* daemon and *stat* daemon.

Missing details

There are some areas in Chapter 10 that need more detail. The section on NFS performance (two paragraphs), should be greatly expanded. More detail is needed on the various mount options. Which combination of hard/soft mounts and interruptibility should be used for various filesystems? What are the implications of UDP checksums on NFS reliability? What about using TCP instead of UDP with NFS across wide area networks?

The final chapter seems out of place: 11 pages on the programmer interface (sockets, XTI, and XDR/RPC). About all that's covered is a list of the functions in each package. There is a short RPC example, but it doesn't even use *rpcgen.*

If you're interested in learning how RPC and NFS operate, you'll enjoy the last half of this book. Whether you can justify the cost of the book for 90 pages, however, is a different question. What I would still like to see is a book just on NFS, with lots of details, assuming the reader already understands TCP/IP.

—*W. Richard Stevens*

Where and how to get new RFCs

by Jon Postel, University of Southern California, Information Sciences Institute

Primary Repositories

Request for Comments (RFCs) can be obtained via FTP from the following primary repositories:

• NIC.DDN.MIL(aka DIIS.DDN.MIL):
RFCs can be obtained via FTP from NIC.DDN.MIL, with the pathname rfc/rfcnnnn.txt(where "nnnn" refers to the number of the RFC). Login with FTP username "anonymous" and password "guest." Contact: ScottW@NIC.DDN.MIL.

• FTP.NISC.SRI.COM:
RFCs can be obtained via FTP from FTP.NISC.SRI.COM, with the pathname rfc/rfcnnnn.txt or rfc/rfcnnnn.ps (where "nnnn" refers to the number of the RFC). Login with FTP username "anonymous" and password "guest." To obtain the RFC Index, use the pathname rfc/rfc-index.txt.

SRI also provides an automatic mail service for sites which cannot use FTP. Address your request to MAIL-SERVER@NISC.SRI.COM and in the body of the message indicate the RFC to be sent: "send rfcNNNN" or "send rfcNNNN.ps" where "NNNN" is the RFC number. Multiple requests may be included in the same message by listing the "send" commands on separate lines. To request the RFC Index, the command should read: "send rfc-index." Contact: April@NISC.SRI.COM.

• NIS.NSF.NET:
To obtain RFCs from NIS.NSF.NET via FTP, login with username "anonymous" and password "guest"; then connect to the RFC directory ("cd RFC"). The file name is of the form RFCnnnn.TXT-1(where "nnnn" refers to the number of the RFC).

The NIS also provides an automatic mail service for those sites which cannot use FTP. Address the request to NIS-INFO@NIS.NSF.NET and leave the subject field of the message blank. The first line of the message body must be "SEND RFCnnnn.TXT-1," where "nnnn" is the RFC number. Contact: Jo_Ann_Ward@UM.CC.UMICH.EDU.

• NISC.JVNC.NET:
RFCs can also be obtained via FTP from NISC.JVNC.NET, with the pathname rfc/RFCnnnn.TXT.v (where "nnnn" refers to the number of the RFC and "v" refers to the version number of the RFC).

JvNCnet also provides a mail service for those sites which cannot use FTP. Address the request to SENDRFC@JVNC.NET and in the subject field of the message indicate the RFC number, as in "Subject: RFCnnnn" where "nnnn" is the RFC number. No text in the body of the message is needed. Contact: Becker@NISC.JVNC.NET.

• VENERA.ISI.EDU:
RFCs can be obtained via FTP from VENERA.ISI.EDU, with the pathname in-notes/rfcnnnn.txt (where "nnnn" refers to the number of the RFC). Login with FTP username "anonymous" and password "guest." Contact: JKRey@ISI.EDU.

• WUARCHIVE.WUSTL.EDU:
RFCs can also be obtained via FTP from WUARCHIVE.WUSTL.EDU, with the pathname info/rfc/rfcnnnn.txt.Z (where "nnnn" refers to the number of the RFC and "Z" indicates that the document is in compressed form).

At WUARCHIVE.WUSTL.EDU the RFCs are in an "archive" file system and various archives can be mounted as part of an NFS file system. Please contact Chris Myers (chris@wugate.wustl.edu) if you want to mount this file system in your NFS.

Secondary Repositories

RFCs can also be obtained from the following secondary repositories:

Sweden

Host:	sunic.sunet.se
Directory:	rfc
Host:	chalmers.se
Directory:	rfc

Germany

Site:	University of Dortmund
Host:	walhalla.informatik.uni-dortmund.de
Directory:	pub/documentation/rfc
Notes:	RFCs in compressed format

France

Site:	Institut National de la Recherche en Informatique et Automatique (INRIA)
Address:	info-server@inria.fr
Notes:	RFCs are available via email to the above address. Info Server manager is Mireille Yamajako (yamajako@inria.fr).

Netherlands

Site:	EUnet
Host:	mcsun.eu.net
Directory:	rfc
Notes:	RFCs in compressed format.

Finland

Site:	FUNET
Host:	funet.fi
Directory:	rfc
Notes:	RFCs in compressed format. Also provides e-mail access by sending mail to archive-server@funet.fi.

Norway

Host:	ugle.unit.no
Directory:	pub/rfc

Denmark

Site:	University of Copenhagen
Host:	ftp.diku.dk (freja.diku.dk)
Directory:	rfc

United States

CSNET Coordination and Information Center
BBN Systems and Technologies Corporation
10 Moulton Street
Cambridge, MA 02238
617-873-2777
INFO@SH.CS.NET

Additional information

Requests for special distribution of RFCs should be addressed to either the author of the RFC in question, to NIC@NIC.DDN.MIL, or to NISC@NISC.SRI.COM.

Submissions for Requests for Comments should be sent to POSTEL@ISI.EDU. Please consult RFC 1111, "Instructions to RFC Authors," for further information.

Requests to be added to or deleted from the RFC distribution list should be sent to RFC-REQUEST@NIC.DDN.MIL.

Changes to the information in this article should be sent to Joyce K. Reynolds (JKRey@ISI.EDU). Hardcopy versions of the RFCs can be purchased from the Network Information Systems Center at SRI International, nisc@nisc.sri.com or 1-415-859-6387.

Announcement and Call for Participation

The *Fifth IEEE Workshop on Metropolitan Area Networks* will be held May 10–13, 1992 at the Hotel Capo Taormina in Taormina, Italy.

Background

Since the first workshop in 1984, metropolitan area networks have evolved from a wide open area with ill defined objectives to a much more mature field with a first generation of network standards for high-speed data services. In addition to having special characteristics, which lie between those of local and wide area networks, the metropolitan area is an important testbed for future broadband and wide area network services and thus provides opportunities for unique solutions to networking problems. The purpose of the present workshop is to examine experience with these technologies and services to determine what problems remain to be solved, and to explore new MAN technologies and their applications in the provisioning of all broadband services.

Topics

Proposal for presentations are solicited that address the following:

- Experience with operational metropolitan and campus networks
- Role of MANs in evolving network hierarchies
- MAN internetworking with LANs, WANs, ISDN, and BISDN
- Gigabit networks
- High speed network protocols: performance and implementations
- Network security requirements and techniques for MANs
- Wireless MANs and base stations with wireless access systems
- Alternative MAN technologies (e.g., spread spectrum, photonic switching, and coherent detection)
- MAN network management and signalling
- Broadband services and multimedia applications

We welcome additional proposals for areas we may have overlooked. All potential attendees are asked to identify what they consider to be the most controversial issue in MANs and state their position. The workshop will have several panel sessions. The workshop committee is considering the possibility of publishing a selection of the best paper submissions after the workshop.

Sending proposals

Please send a one-page summary of your proposed presentation and/or position statement by paper or electronic mail to either program co-chairpersons:

Pitro Zafiropulo	Andres Albanese
IBM Research Division	Bellcore
Zürich Research Laboratory	445 South Street
Sümerstr. 4	Morristown,
8803 Rüschlikon	NJ 07962-1910
Switzerland	USA
Tel.: +41-1-7248-310	+1 (201) 829-4291
Fax: +41-1-7103-608	+1 (201) 829-5888
E-Mail: `pz@ibm.com`	`aa@bellcore.com`

Proposals must be received before January 10, 1992 and notification of acceptance will be sent by February 28, 1992. Attendance will be limited.

Workshop Announcement

The University of Maryland Institute for Advanced Computer Studies (UMIACS) in cooperation with the Computer Science Department at UMBC will hold the *3rd Workshop on Very High Speed Networks* on March 9–10, 1992 at the Greenbelt Marriott Hotel in Greenbelt, MD.

Goal

The goal of the workshop is to bring together experts in related areas to discuss the progress and research issues in the design and implementation of very high speed (gigabit rates) communication networks. Last year's workshop attracted approximately 165 researchers representing academia, industry and government. The meeting will include invited speakers and contributed presentations. Selected papers will appear in a special issue of the *Journal of High Speed Networks.*

Registration

A $125.00 registration fee will include two lunches, a proceedings and conference materials. A limited number of double and single sleeping rooms reserved at the discount rate of $85/night are available at the Greenbelt Marriott for workshop attendees. Room reservations must be made with the hotel by February 17, 1992. The discount rate and room availability are not guaranteed after that date. Contact the Greenbelt Marriott toll free at 1-800-228-9290 or in the Washington D.C. area at 1-301-441-3700. Be sure to identify yourself as a participant of the *UMIACS Workshop on High Speed Networks.*

Preregistration must be received no later than February 17, 1992. If additional information is needed, please contact Dawn Vance at the address below. For questions regarding the technical content of the workshop, please contact the workshop organizer, Deepinder Sidhu, at `sidhu@umbc3.umbc.edu` or 1-301-455-3028.

More information

Dawn Vance
UMIACS Meeting Planner
A.V. Williams Building
University of Maryland
College Park, MD 20742
1-301-405-6730 • `dawn@umiacs.umd.edu`

Greenbelt Marriott
6400 Ivy Lane
Greenbelt, MD 20770
1-301-441-3700
1-800-228-9290

"Components of OSI" in *ConneXions*

Integrated Services Digital Network (ISDN)	April	1989
X.400 Message Handling System	May	1989
X.500 Directory Services	June	1989
The Transport Layer	July	1989
Routing overview	August	1989
IS–IS Intra-Domain Routing	August	1989
ES–IS Routing	August	1989
The Session Service	September	1989
Connectionless Network Protocol (CLNP)	October	1989
The Presentation Layer	November	1989
A taxonomy of the players	December	1989
The Application Layer Structure	January	1990
File Transfer, Access, and Management (FTAM)	April	1990
The Security Architecture	August	1990
Group Communication	September	1990
X.25—the Network, Data Link, & Physical Layers	December	1990
The Virtual Terminal ASE	January	1991
Systems Management	April	1991
CO/CL Interworking	May	1991
Open/Office Document Architecture (ODA)	August	1991
Abstract Syntax Notation One (ASN.1)	January	1992

CONNEXIONS
480 San Antonio Road
Suite 100
Mountain View, CA 94040
415-941-3399
FAX: 415-949-1779

FIRST CLASS MAIL
U.S. POSTAGE
PAID
SAN JOSE, CA
PERMIT NO. 1

ADDRESS CORRECTION
REQUESTED

CONNEXIONS

EDITOR and PUBLISHER	Ole J. Jacobsen
EDITORIAL ADVISORY BOARD	Dr. Vinton G. Cerf, Vice President, Corporation for National Research Initiatives
	A. Lyman Chapin, Chief Network Architect, BBN Communications
	Dr. David D. Clark, Senior Research Scientist, Massachusetts Institute of Technology
	Dr. David L. Mills, Professor, University of Delaware
	Dr. Jonathan B. Postel, Communications Division Director, University of Southern California, Information Sciences Institute

CONNEXIONS

Subscribe to CONNEXIONS

U.S./Canada ❑ $150. for 12 issues/year ❑ $270. for 24 issues/two years ❑ $360. for 36 issues/three years

International $ 50. additional **per year** **(Please apply to all of the above.)**

Name ______________________ Title ______________________

Company ______________________

Address ______________________

City ______________________ State ____________ Zip ____________

Country ______________________ Telephone (   ) ______________________

❑ Check enclosed (in U.S. dollars made payable to CONNEXIONS).
❑ Visa ❑ MasterCard ❑ American Express ❑ Diners Club Card # ______________________ Exp.Date ________

Signature ______________________

Please return this application with payment to: **CONNEXIONS**
480 San Antonio Road, Suite 100
Mountain View, CA 94040 U.S.A.
415-941-3399 FAX: 415-949-1779

Back issues available upon request $15./each
Volume discounts available upon request